

“E pluribus unum”

or

# How to Derive Single-equation Descriptions for Output-quantities in Nonlinear Circuits using Differential Algebra (2008 Re-Release)

Eberhard H.-A. Gerbracht

*Dedicated to the memories of*  
*Prof. Dr.-Ing. Ernst-Helmut Horneber (1946–2001)*  
*and*  
*Prof. Giuseppa Carrà Ferro (1952–2007)*

**Abstract**—In this paper we describe by a number of examples how to deduce one single characterizing higher order differential equation for output quantities of an analog circuit.

In the linear case, we apply basic “symbolic” methods from linear algebra to the system of differential equations which is used to model the analog circuit. For nonlinear circuits and their corresponding nonlinear differential equations, we show how to employ computer algebra tools implemented in Maple, which are based on differential algebra.

**Keywords** (semi-)state equations, systems of nonlinear ODEs, nonlinear circuits, differential algebra, Maple.

**Mathematics Subject Classification (2000)** Primary 34A09; Secondary 12H05, 65W30, 94C05

## I. INTRODUCTION

Usually the input-output response of a linear time-invariant circuit is described in the frequency domain by its transfer function, i.e. a single rational function. This translates directly into a linear differential equation with constant coefficients in the time domain. The advantage of this approach is, that in any guise, only one (differential) equation is needed to completely describe the quantity a designer is interested in.

This method fails miserably, when a transformation from the time to the frequency domain is not possible, e.g., when nonlinear circuits have to be examined, which lead to systems of nonlinear differential equations. Even though most of the times, these can be given in symbolic terms, any single quantity in the circuit usually is described by a “waveform”, which results from assigning a numerical value to each symbol, followed by a numerical calculation using computer simulation. The advantage of having only *one* describing equation seems to have been irrevocably lost, when nonlinear

circuits are considered. Thus, up until now, nonlinear circuits seemed nearly inaccessible to most symbolic approaches.

This problem is not a new one, and it is not limited to the area of analog circuits alone. Several years ago, researchers in nonlinear control theory have proposed to use constructive methods from differential algebra to tackle their problems. At the same time – and inspired in part by this proposal – mathematicians started to implement algorithms from differential algebra, which had already been formulated in the 1950s. These programmes became part of the MAPLE computer algebra system.

At the SMACD-meeting in 1998 G. Carrà Ferro<sup>1</sup> gave examples of how to transform systems of nonlinear differential equations containing certain transcendental functions, that arise from analog circuits, into systems of nonlinear “algebraic” differential equations [1], and thus brought the area of constructive differential algebra in contact with the area of symbolic circuit analysis and design. In this paper we will take this approach several steps further and show, how single equations for any quantity in a circuit can be derived from these systems. We will be able to give a new and easy algorithm for linear circuits, which works in the time domain, and uses only differentiation and Gaussian elimination. For nonlinear circuits we will resort to the algorithms already implemented in MAPLE. We will comment on how to apply them and produce several examples.

## II. LINEAR CIRCUITS AND LINEAR SYSTEMS OF DIFFERENTIAL EQUATIONS

To get a first flavour of things, we start our discussion with linear circuits and their corresponding systems of linear differential equations. But, instead of working in the frequency domain, using the Laplace-transform to deduce the transfer function for a sought-after quantity and thus its characterizing differential equation, we will remain in the time domain.

### A. Linear State Equations

In this section we will present a new algorithm, that, starting from a set of state equations, only uses repeated differentiation and, finally, Gaussian elimination, to compute the single differential equation for any given quantity. We will

This article first appeared in: Proceedings of the 7th International Workshop on Symbolic Methods and Applications to Circuit Design, SMACD 2002, Sinaia, Romania, October 10-11, 2002. Bukarest 2002, pp. 65–70. Due to the low distribution of these proceedings, the author has decided to make the article available to a wider audience through the arXiv.

At the time of origin of this paper the author was with the Institut für Netzwerktheorie und Schaltungstechnik, Technische Universität Braunschweig, D-38106 Braunschweig, Germany.

His current (April 17th, 2008) address is Bismarckstraße 20, D-38518 Gifhorn, Germany. Current e-mail: e.gerbracht@web.de

<sup>1</sup>Note added in 2008: Between the time of this article’s first publication and the publication of the electronic version, Giuseppa Carrà Ferro passed away on March 22nd, 2007. Thus a dedication to her memory was added to this electronic version.

restrict our presentation to just three state variables  $x_1, x_2, x_3$ , but it is easy to extend the method, shown below, to any number of state variables.

So, let us suppose, that a given linear circuit can be described by a set of linear state equations. We ask for one single differential equation describing, without loss of generality, the state variable  $x_1$ . Again it is easy to handle other state variables or any output variable  $y$ , which is a linear combination of state variables and inputs, in an analogous manner.

Let the system be given by

$$\begin{aligned}\dot{x}_1(t) &= a_{11} \cdot x_1(t) + a_{12} \cdot x_2(t) + a_{13} \cdot x_3(t) + e_1(t), \\ \dot{x}_2(t) &= a_{21} \cdot x_1(t) + a_{22} \cdot x_2(t) + a_{23} \cdot x_3(t) + e_2(t), \\ \dot{x}_3(t) &= a_{31} \cdot x_1(t) + a_{32} \cdot x_2(t) + a_{33} \cdot x_3(t) + e_3(t),\end{aligned}\quad (1)$$

where  $x_1, x_2, x_3$  denote the state variables and  $e_1, e_2, e_3$  represent linear combinations of the inputs (and eventually their derivatives).

Clearly, if  $x_1, x_2, x_3$  satisfy (1), their derivatives<sup>2</sup> will satisfy

$$\begin{aligned}\ddot{x}_1(t) &= a_{11} \cdot \dot{x}_1(t) + a_{12} \cdot \dot{x}_2(t) + a_{13} \cdot \dot{x}_3(t) + \dot{e}_1(t), \\ \ddot{x}_2(t) &= a_{21} \cdot \dot{x}_1(t) + a_{22} \cdot \dot{x}_2(t) + a_{23} \cdot \dot{x}_3(t) + \dot{e}_2(t), \\ \ddot{x}_3(t) &= a_{31} \cdot \dot{x}_1(t) + a_{32} \cdot \dot{x}_2(t) + a_{33} \cdot \dot{x}_3(t) + \dot{e}_3(t).\end{aligned}\quad (2)$$

If  $n > 3$  state variables are given, we have to repeat the above procedure  $n - 1$  times. It is a well known observation (which will be shown as a byproduct of our algorithm), that  $n$  linear first order state equations lead to one  $n$ th order differential equation for any single quantity. Thus, in our example, we need one further equation for the third derivative  $\ddot{x}_1$  of  $x_1$ , which we get by differentiating once again the first equation of (2).

$$\ddot{x}_1(t) = a_{11} \cdot \ddot{x}_1(t) + a_{12} \cdot \ddot{x}_2(t) + a_{13} \cdot \ddot{x}_3(t) + \ddot{e}_1(t). \quad (3)$$

Next, we write down all of the above equations into one system, where the vector of variables is given by all the derivatives of all state variables, that have been produced by the above procedure.

$$\begin{pmatrix} -a_{12} & -a_{13} & 1 & -a_{11} & 0 & -a_{21} & 0 & -a_{31} \\ 0 & 0 & -a_{12} & -a_{13} & 1 & -a_{11} & 0 & -a_{21} \\ 1 & 0 & -a_{22} & -a_{23} & 0 & -a_{21} & 1 & -a_{11} \\ 0 & 1 & -a_{32} & -a_{33} & 0 & -a_{31} & 0 & -a_{21} \\ & 0 & 0 & -a_{12} & -a_{13} & 1 & -a_{11} & 0 \\ & 1 & 0 & -a_{22} & -a_{23} & 0 & -a_{21} & 1 \\ & 0 & 1 & -a_{32} & -a_{33} & 0 & -a_{31} & 0 \end{pmatrix} \cdot \begin{pmatrix} \ddot{x}_2(t) \\ \ddot{x}_3(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ x_2(t) \\ x_3(t) \\ \ddot{x}_1(t) \\ \dot{x}_1(t) \\ x_1(t) \end{pmatrix} = \begin{pmatrix} \ddot{e}_1(t) \\ \dot{e}_1(t) \\ \dot{e}_2(t) \\ \dot{e}_3(t) \\ e_1(t) \\ e_2(t) \\ e_3(t) \end{pmatrix}.$$

(4)

We point out the fact, that the variables should be arranged according to the ordering  $\ddot{x}_2(t) > \ddot{x}_3(t) > \dot{x}_2(t) > \dot{x}_3(t) > x_2(t) > x_3(t) > \ddot{x}_1(t) > \dot{x}_1(t) > x_1(t)$ . For our algorithm to work, it is essential, that the last entries of the vector are the derivatives of the variable, for which we want to deduce the differential equation, in decreasing order.

<sup>2</sup>In the sequel wherever necessary, we will assume that all expressions are differentiable over a suitable extension field of the real numbers.

The final step is to use Gaussian elimination to convert the system into one in upper triangular form

$$\begin{pmatrix} 1 & * & * & * & * & * & * & * & * & * \\ & 1 & * & * & * & * & * & * & * & * \\ & & 1 & * & * & * & * & * & * & * \\ & & & 1 & * & * & * & * & * & * \\ & & & & 1 & * & * & * & * & * \\ & & & & & 1 & * & * & * & * \\ & & & & & & 1 & -\alpha_2 & -\alpha_1 & -\alpha_0 \end{pmatrix} \cdot \begin{pmatrix} \ddot{x}_2(t) \\ \ddot{x}_3(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ x_2(t) \\ x_3(t) \\ \ddot{x}_1(t) \\ \dot{x}_1(t) \\ x_1(t) \end{pmatrix} = \begin{pmatrix} * \\ * \\ * \\ * \\ * \\ * \\ * \\ \Psi(t) \end{pmatrix}, \quad (5)$$

where  $\Psi(t)$  is a linear combination of the functions  $\ddot{e}_1(t)$ ,  $\dot{e}_1(t)$ ,  $e_1(t)$ ,  $\dot{e}_2(t)$ ,  $e_2(t)$ ,  $\dot{e}_3(t)$ , and  $e_3(t)$ .

Since we had  $(n-1) \cdot n + 1$  (obviously) linearly independent equations for  $n^2 + 1$  variables, the last row will give one equation for the last  $n + 1$  variables, which according to our special ordering are all derivatives of  $x_1$ . Thus we have deduced the differential equation for  $x_1$ , which we were looking for. This is the same equation, which we would have found, if we had worked in the frequency domain and had calculated the transfer function for the Laplace transform  $\mathcal{L}(x_1)$  of  $x_1$ . ■

## B. Linear SemiState Equations

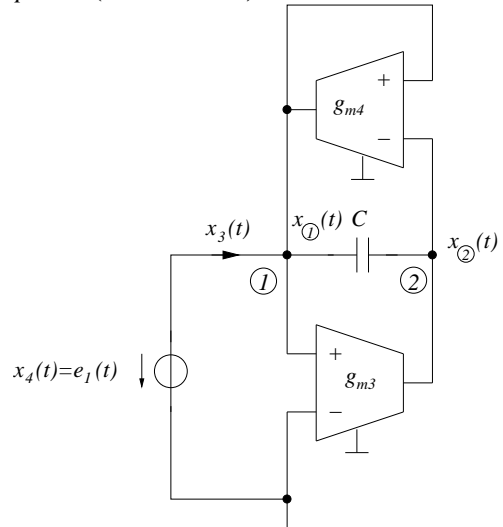
It might happen, that a linear time-invariant circuit does not possess a description by state equations. Nevertheless it may be describable by so-called semistate equations, i.e. equations of the form

$$\mathbf{E} \dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{B} \mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C} \mathbf{x}(t) + \mathbf{D} \mathbf{u}(t), \quad (6)$$

in which  $\mathbf{A}$ ,  $\mathbf{B}$ ,  $\mathbf{C}$ ,  $\mathbf{D}$  and  $\mathbf{E}$  are constant matrices,  $\mathbf{E}$  being singular,  $\mathbf{x}(t)$  denotes the semistate vector,  $\mathbf{u}(t)$  the vector of inputs and  $\mathbf{y}(t)$  the vector of outputs.

The algorithm given above can be adapted to this situation, but we have to keep in mind, that even if  $n$  semistate equations are given, the differential equation for any quantity might be of degree strictly less than  $n$ . We will see this effect in the example given below, which recently appeared in [2].

### Example 1: (OTA-Circuit)



The OTA circuit shown above can be described by the semi-state system

$$\begin{pmatrix} C & -C & 0 \\ -C & C & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_{\textcircled{1}}(t) \\ \dot{x}_{\textcircled{2}}(t) \\ \dot{x}_3(t) \end{pmatrix} = \begin{pmatrix} -g_{m4} & g_{m4} & 1 \\ -g_{m3} & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_{\textcircled{1}}(t) \\ x_{\textcircled{2}}(t) \\ x_3(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -e_1(t) \end{pmatrix},$$

where  $x_{\textcircled{1}}(t)$  and  $x_{\textcircled{2}}(t)$  denote the node to ground voltages of the respective nodes and  $x_3(t)$  denotes the current into node  $\textcircled{1}$ .

As shown in [2], the transfer function  $\underline{H}(s) := \frac{X_3(s)}{E_1(s)}$  is given by

$$\underline{H}(s) = g_{m3} \frac{Cs + g_{m4}}{Cs}. \quad (7)$$

We will deduce the corresponding differential equation for  $x_3(t)$  in the time domain, using the above system of semistate equations and a slight modification of the algorithm for state equations. This algorithm is closer to the one, that will be used for nonlinear circuits. It is based on two main principles:

- 1) The variables  $x_{\textcircled{1}}$ ,  $x_{\textcircled{2}}$ ,  $x_3$  and their derivatives, are supposed to be ordered by

$$\begin{aligned} x_3 < \dot{x}_3 < \ddot{x}_3 < \dots < x_{\textcircled{2}} < \dot{x}_{\textcircled{2}} < \ddot{x}_{\textcircled{2}} < \dots \\ < x_{\textcircled{1}} < \dot{x}_{\textcircled{1}} < \ddot{x}_{\textcircled{1}} < \dots \end{aligned}$$

When a term is to be eliminated, we always choose the term of highest order.

- 2) Equations are only differentiated when "necessary".

So, let us get into details; the system was given by

$$C \dot{x}_{\textcircled{1}} - C \dot{x}_{\textcircled{2}} = -g_{m4} x_{\textcircled{1}} + g_{m4} x_{\textcircled{2}} + x_3 \quad (8)$$

$$-C \dot{x}_{\textcircled{1}} + C \dot{x}_{\textcircled{2}} = -g_{m3} x_{\textcircled{1}} \quad (9)$$

$$0 = x_{\textcircled{1}} - e_1 \quad (10)$$

The term of highest order appearing in (8) - (10) is  $x_{\textcircled{1}}$ . Equation (10) gives

$$x_{\textcircled{1}} = e_1 \quad (11)$$

This can be used to eliminate  $x_{\textcircled{1}}$  from (8) and (9). Furthermore, after differentiating (11), we get

$$\dot{x}_{\textcircled{1}} = \dot{e}_1. \quad (12)$$

Thus we are able to remove all instances of  $x_{\textcircled{1}}$  in the first two equations. Consequently we are led to:

$$C \dot{x}_{\textcircled{2}} = C \dot{e}_1 - g_{m3} e_1 \quad (13)$$

$$C \dot{x}_{\textcircled{2}} = C \dot{e}_1 + g_{m4} e_1 - g_{m4} x_{\textcircled{2}} - x_3 \quad (14)$$

Clearly, these two equations imply the equality

$$-g_{m3} e_1 = g_{m4} e_1 - g_{m4} x_{\textcircled{2}} - x_3, \quad (15)$$

which gives

$$x_{\textcircled{2}} = \frac{g_{m3} + g_{m4}}{g_{m4}} e_1 - \frac{x_3}{g_{m4}} \quad (16)$$

and

$$\dot{x}_{\textcircled{2}} = \frac{g_{m3} + g_{m4}}{g_{m4}} \dot{e}_1 - \frac{\dot{x}_3}{g_{m4}} \quad (17)$$

Putting this into (14), we arrive at

$$C \dot{e}_1 - g_{m3} e_1 = \frac{C}{g_{m4}} \cdot (g_{m3} + g_{m4}) \dot{e}_1 - C \cdot \frac{\dot{x}_3}{g_{m4}}, \quad (18)$$

which finally results in

$$C \cdot \dot{x}_3 = C \cdot g_{m3} \cdot \dot{e}_1 + g_{m3} \cdot g_{m4} \cdot e_1. \quad (19)$$

This is the time domain equivalent of the transfer function (7), which we wanted to deduce. ■

### III. CONSTRUCTIVE DIFFERENTIAL ALGEBRA AND THE diffalg-PACKAGE IN maple

This is not the time and the space to give even a cursory treatment of those parts of differential algebra, which are needed to understand the sometimes subtle generalization to the nonlinear case of the algorithms shown above. For our purposes it is enough to know, that, cum grano salis, all the mechanisms are already visible in the example of the OTA circuit.

Fortunately there already exist several implementations of the necessary algorithms within the computer algebra system MAPLE. We will show by way of the above example the workings of one of these, the diffalg-package, created by F. Boulier [3], [4] and improved by E. Hubert [5] et al. A more detailed description (suitable for beginners with a mathematical background) can be found in the world wide web [6], [7].

After starting a MAPLE-session, one first has to load the package diffalg:

```
> with(diffalg);
[Rosenfeld_Groebner, belongs_to, delta_leader, ...]
```

(The second line in slanted notation represents the output produced by diffalg.)

After initialization, one has to enter the set of differential equations under consideration. This has to be done in form of so-called *differential polynomials*. These are polynomials in the unknown functions  $x_1, \dots, x_m$ , their (time) derivatives  $x_i^{(\alpha)} := D^{(\alpha)} x_i := \frac{d^\alpha}{dt^\alpha} x_i$ ,  $1 \leq i \leq m$ ,  $\alpha \in \mathbb{N}$ , the excitations  $e_1, \dots, e_k$  and their derivatives, again.

In our example, we get three differential polynomials, which read in MAPLE-notation:

```
> p_1 := C*diff(x_1(t),t)-C*diff(x_2(t),t)+g_m4*
x_1(t)-g_m4*x_2(t)-x_3(t);
> p_2 := -C*diff(x_1(t),t)+C*diff(x_2(t),t)+
g_m3*x_1(t);
> p_3 := x_1(t)-e_1(t);
```

Next we have to tell the programme, which symbols it has to treat as constants. This is done with the command

field\_extension.difffalg assumes, that we work over the rational numbers as ground field, where any further constants are considered as lying in a transcendental field extension of  $\mathbb{Q}$  (i.e. we are allowed to divide by constants different from 0, and constants do not satisfy any algebraic relations). If we work with symbols, e.g. for capacitors, resistors etc., this poses no problem. If we work with real coefficients (e.g. floating point numbers or algebraic numbers like  $\sqrt{2}$ ) major problems may arise. In our case we define

```
> K := field_extension(transcendental_elements =
[C, g_m3, g_m4]);
```

```
K := ground_field
```

Finally we define a so-called *differential ring*, which is supposed to contain all the objects of interest (i.e. differential polynomials and constants), and in which we are allowed to do the following operations

- 1) multiply a differential polynomials with a constant;
- 2) add and multiply differential polynomials;
- 3) differentiate differential polynomials (if a constant is differentiated, the result is 0).

As we have seen above, it is very important, that we define an ordering on differential monomials. This is needed to control the elimination process. For this purpose, difffalg asks for a ranking of the time dependent variables, from which it produces the obvious "elimination ordering". The variable, for which we want to know the differential equation, should be the last before the excitations.

```
> R := differential_ring(ranking = [x_1, x_2, x_3, e_1],
derivations = [t], field_of_constants = K,
notation = diff);
```

```
R := ODE_ring
```

The command Rosenfeld\_Groebner lies at the heart of difffalg. It produces minimal sets of differential polynomials generating the differential polynomials, we have entered.

```
> GE := Rosenfeld_Groebner({p_1, p_2, p_3}, R);
```

```
GE := [regular]
```

GE is a list and may contain several components<sup>3</sup>. These components contain the sought-after differential equations, which can be listed with the help of the rewrite\_rules command; in our example this gives

```
> rewrite_rules(GE[1]);
```

$$\left[ \begin{array}{l} x_1(t) = e_1(t), \\ x_2(t) = \frac{-x_3(t) + e_1(t)g_{m3} + e_1(t)g_{m4}}{g_{m4}}, \\ \frac{\partial}{\partial t}x_3(t) = \frac{g_{m3} \left( e_1(t)g_{m4} + C \left( \frac{\partial}{\partial t}e_1(t) \right) \right)}{C} \end{array} \right]$$

Mark, that the last entry is the differential equation for  $x_3$ , which we had deduced in example 1.

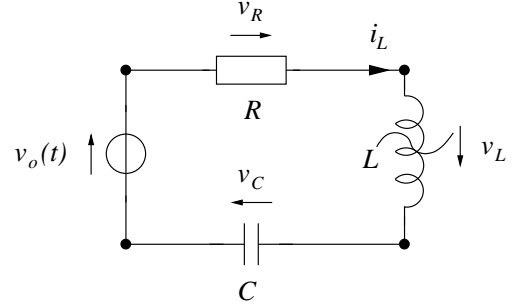
<sup>3</sup>This is one of the subtleties of working with difffalg, the details of which we do not want to present here.

#### IV. NONLINEAR CIRCUITS AND THEIR CORRESPONDING DIFFERENTIAL EQUATIONS

Now, that we know, how the MAPLE package difffalg can be used to deduce the time equivalent of the transfer function from any system of linear differential equations (with constant coefficients), in this section we will apply difffalg in an analogous manner to several systems of nonlinear equations, which come from nonlinear circuits.

##### Example 2: (Damped resonant circuit with a nonlinear inductivity)

As our first nonlinear example we have chosen a well known circuit from the literature, which leads to the Duffing equation (cp. [8], example 11-1, or [9], chapter 1.3.2.).



The resistor and the capacitor are assumed to be linear, i.e., they are described by

$$i_C(t) = C\dot{v}_C(t) \quad \text{and} \quad v_R(t) = Ri_L(t). \quad (20)$$

The inductor is assumed to be nonlinear, being described by

$$v_L(t) = \dot{\Psi}(t), \quad (21)$$

where the current  $i_L(t)$  is approximated by the cubic

$$i_L(t) = a \cdot \psi(t) + b \cdot \psi(t)^3. \quad (22)$$

Finally, Kirchhoff's equation lead to

$$i_R(t) = i_L(t) = i_C(t) \quad \text{and} \quad v_R(t) + v_L(t) + v_C(t) = -v_o(t). \quad (23)$$

Equations (20)-(23) are translated into their corresponding differential polynomials and are used as input for the difffalg-routine. This produces as part of the output of the subroutine rewrite\_rules:

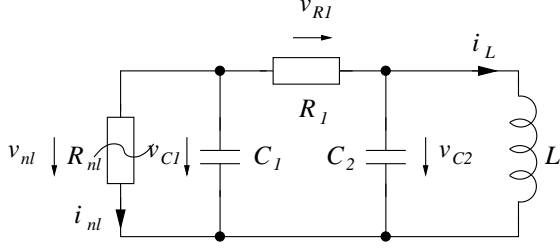
$$\frac{\partial^2}{\partial t^2}\psi(t) = - \frac{a\psi(t) + b\psi(t)^3 + \left(\frac{\partial}{\partial t}v_o(t)\right)C}{C} - \frac{CRa \left(\frac{\partial}{\partial t}\psi(t)\right) + 3CRb\psi(t)^2 \left(\frac{\partial}{\partial t}\psi(t)\right)}{C}, \quad (24)$$

which is formula 1.30 in [9] and is an equivalent of the Duffing equation:

$$\frac{d^2}{dt^2}\psi = -(a + 3b \cdot \psi^2) \cdot R \frac{d}{dt}\psi - \frac{a\psi}{C} - \frac{b\psi^3}{C} - \frac{d}{dt}v_o(t) \quad (25) \quad \blacksquare$$

**Example 3: (Chua's Circuit)** Our next example is Chua's circuit, the first example of a physically realizable circuit,

showing chaotic behavior [10]. This circuit consists of two linear capacitors, one linear inductor, one linear resistor and one nonlinear resistor, as shown below.



Since it is easy to write down the equations for the linear elements and the Kirchhoff equations, we concentrate on the mathematical description of the nonlinear resistor. The classic description of  $R_{nl}$ , using piecewise linear functions, is unsuitable for our purposes, because it does not satisfy the right differentiability conditions. Thus we use the one, presented e.g. in [11], where the negative arctangent is used to produce the nonlinear  $v$ - $i$  characteristic of  $R_{nl}$ . This leads to

$$i_{nl}(t) = -I_0 \cdot \arctan\left(\frac{v_{nl}(t)}{V_0}\right). \quad (26)$$

We are now confronted with a new problem: the arctangent is a transcendental function, thus (26) would not give a differential polynomial, as needed. One procedural solution to this problem was presented before in [1]. In the present paper, we are satisfied with the fact, that (26) implies yet another algebraic differential equation

$$\frac{d}{dt}i_{nl}(t) = -\frac{I_0}{V_0} \cdot \left(1 + \left(\frac{v_{nl}(t)}{V_0}\right)^2\right)^{-1} \cdot \frac{d}{dt}v_{nl}(t) \quad (27)$$

which obviously is given by a differential polynomial and which we can use as input for `diffalg` instead of (26). From this equation together with the Kirchhoff equations and the characterizing equations of the linear elements, `diffalg` produces the differential equation

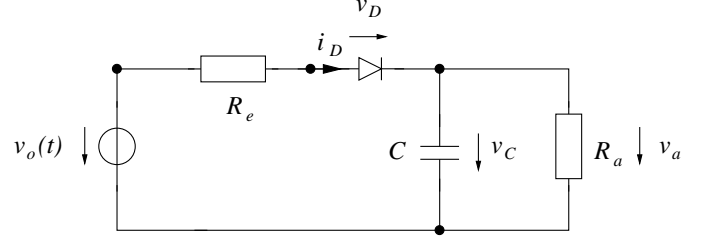
$$\begin{aligned} x^{(4)} = & -\frac{1}{C_1 C_2 R L} \cdot \left( (C_1 + C_2) L \ddot{x} + C_1 R \dot{x} + \dot{x} \right. \\ & - I_0 \cdot \frac{V_0}{x^2 + V_0^2} [C_2 L R \ddot{x} + L \ddot{x} + R \dot{x}] \\ & + 2 I_0 \cdot \frac{V_0}{(x^2 + V_0^2)^2} \cdot L x \dot{x} \cdot [3 C_2 R \ddot{x} + \dot{x}] \\ & \left. - I_0 \cdot \frac{V_0 (6x^2 - 2V_0^2)}{(x^2 + V_0^2)^3} C_2 L R \dot{x}^3 \right), \end{aligned} \quad (28)$$

where  $x(t)$  denotes the voltage  $v_{C1}(t)$  through the capacitor  $C_1$ . It has to be said that the final result given by `diffalg` looks slightly different, since it is given in expanded form, i.e. the numerator consists of 31 summands. Formula (28) has been reached at, only after some laborious post-processing. ■

#### Example 4: (Simple Model of a Peak Rectifier Circuit)

Now we are going to show how to handle diodes (and consequently by way of the Ebers-Moll model the large signal

behaviour of BJTs) in electric circuits. As an example we have chosen a simple model of a peak rectifier circuit as seen in [12], chapter 3.7 pp. 185ff.



Again we concentrate on the only nonlinear element in the circuit – the diode. It is well known, that the  $v$ - $i$  characteristic of a nonideal diode can be approximately described by

$$i_D(t) = I_s \cdot \left[ \exp\left(\frac{v_D(t)}{V_T}\right) - 1 \right], \quad (29)$$

where  $I_s$  is the saturation current and  $V_T$  is the thermal voltage – quantities, which we consider constant during the course of our analysis.

As before we have to translate a transcendental equation into a differential polynomial. This can be done easily by differentiating (29) once, which due to the chain rule  $\frac{d}{dt}i_D(t) = \frac{d}{dv_D}i_D \cdot \frac{d}{dt}v_D(t)$  results in the equation

$$\frac{d}{dt}i_D(t) = \frac{1}{V_T} \left( \frac{d}{dt}v_D(t) \right) \cdot [i_D(t) + I_s]. \quad (30)$$

This time `diffalg` produces the following second order differential equation for the output voltage  $v_a(t)$ :

$$\ddot{v}_a = -\frac{R_a \cdot [V_T \dot{v}_a - (\dot{v}_0 - \dot{v}_a) \cdot v_x] + [\dot{v}_a \cdot R_e v_x]}{C R_a \cdot (V_T R_a + R_e v_x)}, \quad (31)$$

where we have set  $v_x := (C R_a \dot{v}_a + v_a + R_a I_s)$ .

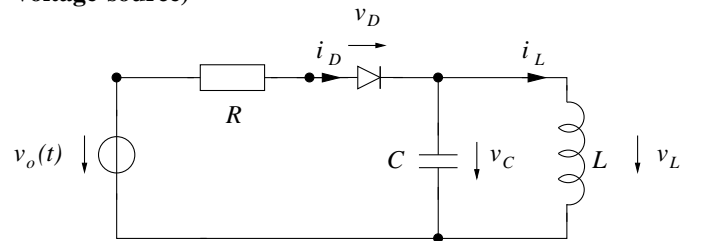
For the purpose of comparing this result to that appearing in the literature, we give the differential equation in case of an ideal voltage source, i.e.  $R_e = 0\Omega$ . It is given by

$$\ddot{v}_a = \frac{1}{C R_a} \left( -\dot{v}_a + \frac{1}{V_T} [\dot{v}_0 - \dot{v}_a] \cdot [C R_a \dot{v}_a + v_a + R_a I_s] \right). \quad (32)$$

■

## V. FURTHER EXAMPLES

### Example 5: (Diode Circuit with LC-Load and Nonideal Voltage source)



The above circuit may not be of much practical interest. Nevertheless it is a good test of the power of our approach (and the capabilities of `diffalg`), since it slightly generalizes example 4 and we increase the number of dynamic elements in our circuit. Again, the diode is assumed to be nonideal, given by (30).

The output voltage  $x(t) := v_C(t)$  is described by the equation

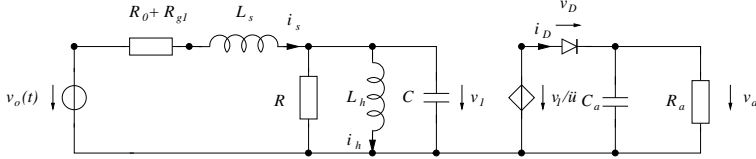
$$\ddot{x} = - \left\{ \frac{1}{CL} \dot{x} + \frac{x + CL\ddot{x}}{\dot{x} - \dot{v}_0} \cdot \frac{1}{V_T \cdot CL} \cdot \left[ \left( \frac{R}{L} (x + CL\ddot{x}) + (\dot{x} - \dot{v}_0) \right)^2 - V_T (\ddot{x} - \ddot{v}_0) \right] \right\}. \quad (33)$$

In the course of our investigations, we have tried a number of larger circuits, which in principle are accessible to our approach. We met two main obstacles, which are natural in the "business" of symbolic methods:

- 1) the combinatorial explosion, resulting in a larger and larger number of terms contained in the final differential equation, and
- 2) the massive increase in time, needed by `diffalg` to produce this equation.

In the sequel we give a short report on these experiments:

**Example 6: (Peak Rectifier with Power Transformer)**



This kind of circuit is described in the introduction to chapter 3.7 in [12]. As shown above, we have used the model given by Horneber in his PhD-thesis [13], section 14.1. There, it is the smaller of two examples, the other being the "Ring Modulator", which has become a benchmark in the numerical analysis of initial value problems [14].

From our point of view, we can tell, that `diffalg`, although needing substantial more time than in the previous examples (1-2 minutes instead of only seconds), is able to produce a fifth order differential equation for the output voltage  $v_a$ . Unfortunately, we are not able to reproduce this result here, since the initial output even after some simplifications contains more than 600 summands. Thus some more "post processing" is needed to get an intelligible result. Even with the help of other facilities of MAPLE this work has not been finished, yet<sup>4</sup>.

Finally we have tried our approach on a "simple" single-stage common-emitter amplifier ([12], chapter 4.11) as modelled in [15] and on the above mentioned ring modulator of Horneber. In both cases, up until now, even though we have used several days of computing time, we were not able to produce any results. Although the latter – very ambitious – example (which presumably will lead to an differential equation of order 18) may be beyond the scope of any computer algebra system for some time, the first should be within our grasp and should be attacked further.

<sup>4</sup>Note added in 2008: Meanwhile these calculations have been done. The end result still is to unwieldy to be presented here. Furthermore the collecting and combining of fully symbolic terms by hand has turned out to be so error-prone that, in the opinion of the author, some kind of additional "plausibility measures" need to be introduced.

## VI. CONCLUSION

In this paper we have shown, how, using constructive methods from differential algebra and one of their realizations – the package `diffalg` of the computer algebra system MAPLE – linear and nonlinear circuits can be described by a single differential equation. In the future it will be necessary to further examine the power of this approach, i.e. to find more and larger circuits, which can be treated this way. Furthermore, if the number of these circuits is large enough, methods have to be found, that allow a fast and "easy" analysis of the resulting equations, analogous to the analysis of linear circuits by way of their transfer functions.

## REFERENCES

- [1] Giuseppa Carrà Ferro, "Computer Algebra, Nonlinear Differential Equations and Analog Circuit Design," in *Proceedings of the 5th International Workshop on Symbolic Methods and Applications in Circuit Design - SMACD '98, Kaiserslautern, October 8-9, 1998*, Dieter Prätzel-Wolters, Ralf Sommer, and Eckhard Hennig, Eds., Kaiserslautern, 1998, pp. 49–56.
- [2] Angela M. Hodge and Robert W. Newcomb, "Semistate Theory and Analog VLSI Design," *IEEE Circuits and Systems Magazine*, vol. 2, no. 2, pp. 30–51, Second Quarter 2002.
- [3] F. Boulrier, D. Lazard, F. Ollivier, and M. Petitot, "Representation for the radical of a finitely generated differential ideal," in *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation, ISSAC '95, Montreal, Canada, July 10–12, 1995*, A.H.M. Levelt, Ed., New York, 1995, pp. 158–166, ACM Press, <https://hal.archives-ouvertes.fr/hal-00138020/en>, viewed April 17th, 2008.
- [4] François Boulrier, *Etude et implantation de quelques algorithmes en algèbre différentielle*, Ph.D. thesis, Université des Sciences et Technologies de Lille, 1994, <http://tel.archives-ouvertes.fr/tel-00137866/en>, viewed April 17th, 2008.
- [5] Evelyne Hubert, *Étude Algébrique et Algorithmique des Singularités des Equations Différentielles Implicites*, Ph.D. thesis, Institut National Polytechnique Grenoble, 1997, <http://www-sop.inria.fr/cafe/Evelyne.Hubert/Publi/EHubertThesis.ps.gz>, viewed April 17th, 2008.
- [6] Evelyne Hubert, "The `diffalg` package," <http://www-sop.inria.fr/cafe/Evelyne.Hubert/diffalg>, viewed April 17th, 2008.
- [7] Evelyne Hubert, "Basic concepts in constructive differential algebra and their representation in the `diffalg` package," [http://www-sop.inria.fr/cafe/Evelyne.Hubert/diffalg/diffalg/differential\\_algebra.html](http://www-sop.inria.fr/cafe/Evelyne.Hubert/diffalg/diffalg/differential_algebra.html), viewed April 17th, 2008.
- [8] Thomas E. Stern, *Theory of Nonlinear Networks and Systems. An Introduction*, Addison-Wesley, Reading, Massachusetts, 1965.

- [9] Eugen S. Philippow and Wolfgang G. Buntig, *Analyse nichtlinearer dynamischer Systeme der Elektrotechnik*, Hanser-Verlag, München, 1992.
- [10] Leon O. Chua, "The Genesis of Chua's Circuit," *Int. J. Electron. Commun. (AEÜ)*, vol. 46, no. 4, pp. 250–257, 1992.
- [11] Thomas Halfmann, Eckhard Hennig, and Manfred Thole, "Behavioral Modeling and Transient Analysis with Analog Insydes," in *Proceedings of the 5th International Workshop on Symbolic Methods and Applications in Circuit Design - SMACD '98, Kaiserslautern, October 8-9, 1998*, Dieter Prätzel-Wolters, Ralf Sommer, and Eckhard Hennig, Eds., Kaiserslautern, 1998, pp. 49–56.
- [12] Adel S. Sedra and Kenneth C. Smith, *Microelectronic Circuits*, Oxford University Press, New York, <sup>4</sup>1998.
- [13] Ernst-Helmut Horneber, *Analyse nichtlinearer RLC-Netzwerke mit Hilfe der gemischten Potentialfunktion mit einer systematischen Darstellung der Analyse nichtlinearer dynamischer Netzwerke*, Ph.D. thesis, Fachbereich Elektrotechnik der Universität Kaiserslautern, 1976.
- [14] Walter M. Lioen and Jacques J.B. de Swart, "Test set for initial value problem solvers," <http://db.cwi.nl/rapporten/abstract.php?abstractnr=796>, 1999, Release 2.1. Also available as <http://ftp.cwi.nl/IVPtestset/>, viewed April 17th, 2008.
- [15] Eckhard Hennig and Ralf Sommer, "Symbolic Methods in Analog Circuit Design," in *2nd IMACS Conference on Applications of Computer Algebra, Linz 17-20 July 1996*, 1996.

#### NOTE ADDED TO THE ELECTRONIC VERSION

In this electronic document, some small typographical errors of the printed version were corrected. This especially refers to formulas (18) and (19).

Furthermore, for the convenience of the reader the abstract has been rewritten, and keywords, an MSC classification, and a short CV according to IEEE standards have been added. URLs have been checked again, and, where necessary, have been updated. Finally the dedication has been expanded. The main body of the article, however, remains unchanged.  
(April 17th, 2008)



**Eberhard H.-A. Gerbracht** received a Dipl.-Math. degree in mathematics, a Dipl.-Inform. degree in computer science, and a Ph.D. (Dr. rer.nat.) degree in mathematics from the Technical University Braunschweig, Germany, in 1990, 1993, and 1998, respectively.

From 1992 to 1997 he was a Research Fellow and Teaching Assistant at the Institute for Geometry at the TU Braunschweig. From 1997 to 2003 he was an Assistant Professor in the Department of Electrical Engineering and Information Technology at the TU Braunschweig. During that time he was also appointed lecturer for several courses on digital circuit design at the University of Applied Sciences Braunschweig/Wolfenbüttel, Germany. From 2001 to 2002 he was appointed lecturer for a two-semester course in linear circuit analysis at the TU Braunschweig. After a two-year stint as a mathematics and computer science teacher at a grammar school in Braunschweig and a vocational school in Gifhorn, Germany, he is currently working as advisor, and independent researcher in various areas of mathematics. His research interests include combinatorial algebra,  $C^*$ -algebras, the history of mathematics in the 19th and early 20th century and applications of computer algebra and dynamical geometry to graph theory, calculus, and electrical engineering.

Dr. Gerbracht is a member of the German Mathematical Society (DMV), the German Society for Didactics of Mathematics (GDM), and founding member of the society "Web Portal: History in Braunschweig - [www.gibs.info](http://www.gibs.info)".